AFRL-RI-RS-TR-2011-148

THE APPLICATION OF COGNITIVE RADIO TO COORDINATED UNMANNED AERIAL VEHICLE (UAV) MISSIONS

VIRGINIA POLYTECHNIC INSTITUTE & STATE UNIVERSITY
VIRGINIA TECH

JUNE 2011

FINAL TECHNICAL REPORT

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■**UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2011-148 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR:

/s/
ROBERT KAMINSKI
Work Unit Manager

/s/
WARREN H. DEBANY, JR., Technical Advisor
Information Grid Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JUNE 2011	Final Technical Report	MAR 2009 – DEC 2010

4. TITLE AND SUBTITLE

THE APPLICATION OF COGNITIVE RADIO TO COORDINATED UNMANNED AERIAL VEHICLE (UAV) MISSIONS

5a. CONTRACT NUMBER
FA8750-09-2-0128

5b. GRANT NUMBER
N/A

5c. PROGRAM ELEMENT NUMBER
62788F

6. AUTHOR(S)

Charles W. Bostian
Alexander R. Young

5d. PROJECT NUMBER
4519

5e. TASK NUMBER
VT

5f. WORK UNIT NUMBER
EC

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Virginia Polytechnic Institute & State University Virginia Tech
1880 Pratt Dr., Ste 2006
Blacksburg, VA 24060-6750

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory/Information Directorate
Rome Research Site/RIG
525 Brooks Road
Rome NY 13441

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RI

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-RI-RS-TR-2011-148

12. DISTRIBUTION AVAILABILITY STATEMENT
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This effort investigated the application of cognitive radio techniques to coordinated Unmanned Aerial Vehicle (UAV) missions. The goal was to explore the potential for the Air Force (a) to replace a UAV's multiple radio platforms with one or more waveform and frequency agile multichannel software defined radios that use cognitive techniques to optimize radio links under changing channel conditions and evolving mission requirements and (b) to employ cooperative communications technique to enhance the performance of UAV networks.

15. SUBJECT TERMS
Cognitive Radio, Cognitive Networks, Small UAV communications

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			ROBERT KAMINSKI
U	U	U	UU	33	19b. TELEPHONE NUMBER (*Include area code*) N/A

TABLE OF CONTENTS

1. Executive Summary 1
2. Introduction 3
2.1 Project Objective 3
2.2 Overview of Cognitive Radio, Cooperative Relaying, and Cooperative Communications Literature 3
2.2.1 Cognitive Radio 3
2.2.2 Cooperative Communications 3
2.2.3 Network Coding 4
2.2.4 Applications to this Project 5
2.3 Potential Advantages of Cognitive Radio in UAV Downlinks 5
3. Methods, Assumptions, and Procedures 6
3.1 Operational Assumptions 6
3.2 Use Case: Cooperative Communications for Small UAVs 6
3.3 Methods 7
3.3.1 Partitioning the Problem 7
3.3.2 Simulating Cognitive Radio Behavior in Real UAV Downlinks 7
3.3.2.1 Simulator Design 7
3.3.2.2 Radio Platform Tests 8
3.3.2.3 Incorporating Measured Data 8
3.3.3 The VT Test Network 9
3.3.3.1 Design 9
3.3.3.2 Operation 10
3.3.3.3 Implementation Details 10
3.3.3.3.1 Overview of the Network 10
3.3.3.3 2 UAV Node 11
3.3.3.3.4 Socket Server 14
3.3.3.3.5 Vision Module 15
3.3.3.3.6 RF Module 18

3.3.3.3.7 Message Handler 20
3.3.3.3.8 Knowledge Base 21
3.3.3.9 Earth Station 22
3.3.3.10 Headquarters 23
4. Results and Discussion 23
4.1 Radio Platforms 23
4.2 Network Simulations 24
5. Conclusions 25
6. Recommendations 26
7. References 26

LIST OF FIGURES

Figure 1 GNU Radio/USRP-Based UAV Downlink Simulator 7
Figure 2 Block diagram representing transmitter and receiver setup and node interconnect configuration for USRP and USRP2 performance tests. 8
Figure 3 CWT AFRL UAV network Implementation. 10
Figure 4 Captured frames displayed on screen of headquarters system. 11
Figure 5 UAV node system architecture diagram. 12
Figure 6 UAV controller architecture. 13
Figure 7 UAV controller primary loop flow. 13
Figure 8 Screen capture showing image identification using OpenCV library 15
Figure 9 Diagram showing program logic and flow for vision subsystem. 17
Figure 10 The RF module's 3-level stack design. 18
Figure 11 Preparing image frames for transport over RF downlink. 19
Figure 12 Packet structure for data used in RF link. 20
Figure 13 Close up view of headquarters display showing images captured by two different UAV nodes displayed in separate windows. 23
Figure 14 UAV network running tests with 6 nodes. This image shows the 6 nodes in operation. 25
Figure 15 UAV network running tests with 6 nodes. This image shows the USRP RF front ends used. 25

LIST OF TABLES

Table 1 System flags in knowledge base. 21
Table 2 Data entries in knowledge base. 22
Table 3 Experimental results using USRP 400 MHz daughter board. 24
Table 4 Summary of system testing. 25

1. Executive Summary

The objective of this project is to investigate the application of cognitive radio techniques to coordinated Unmanned Aerial Vehicle (UAV) missions. We intend the project to be the first phase of a three-year effort to develop low-cost cognitive radios for deployment in small UAVs. Its goal is to explore the potential for the Air Force (a) to replace a UAV's multiple radio platforms with one or more waveform and frequency agile multichannel software defined radios that use cognitive techniques to optimize radio links under changing channel conditions and evolving mission requirements and (b) to employ cooperative communications technique to enhance the performance of UAV networks. Our work focuses on small UAV platforms flown by the USAFRL. We have concentrated on what we think can be achieved in practice using near-term available hardware and software that is compatible with the small UAV platforms of interest.

After reviewing several approaches to cooperative communications, we chose and implemented a design that incorporates elements of the decode and forward (DF) method together with a routing algorithm that configures the radios in a UAV network to give maximum priority and channel capacity to the aircraft with a high priority target. It allows all members of the network to share the downlinking burden. We envision a scenario in which a coordinated group of UAVs is searching for targets. Each UAV has its own downlink to an earth station for transmitting target data and the UAVs are also able to share information over their own ad hoc radio network. The downlinks operate with cognitive radios which attempt to adjust their operating parameters (transmitter power, modulation type and index, operating frequency, data rate, etc.,) as necessary to maintain the acceptable link quality given operational constraints like available battery power. Under ideal conditions each UAV is able to downlink its own target data in real time without needing recourse to the other aircraft. But when a UAV with a high-value target is unable to maintain the required link quality of service on its own downlink, it is able to cooperate with the other aircraft to get its target information "home."

Partitioning the problem into a radio problem and a routing problem, we investigated the capability of cognitive radios based on the GNU Radio/Universal Software Radio Peripheral (USRP) open-source platform to meet the needs of the UAVs of interest. Our tests indicated that the GNU Radio/USRP combination can support data rates of at least 1 Mbps for small UAV platforms using DBPSK, DQPSK, and GMSK modulations in the 300-500 MHz frequency range. We built a simulator that would allow testing the radios with USAFRL measured data, but due to delays in gaining access to the data and the software necessary to manipulate it, we were unable to complete this work. We plan to do it in a subsequent project when we will test a flight version of a cognitive radio for small UAVs.

To test the routing algorithm in conjunction with GNU Radio/USRP cognitive radio platforms, we built a network consisting of up to five nodes representing UAVs (We call these "UAV nodes") and a node representing an earth stations ("E/S node"). The uplinks and downlinks all operate in the 450 MHz FRS band, 462.5625 to 467.7125 MHz, whose characteristics are very similar to those of the military 310-390 MHz band. The UAV nodes form an ad hoc WiFi network to communicate among themselves. Every node in the UAV network takes pictures with its attached camera at a rate of X frames per second (FPS).

These frames are forwarded over the ad-hoc network to the edge. The edge then transmits the frames to the E/S, which forwards it to a central location for display. If a UAV node identifies a target in a particular frame it has captured, it 1) notifies all the nodes in the network that it has acquired a target, and 2) increases its capture rate to 10X. Upon receiving notification that another node has identified a target, a node reduces its own capture rate to Y (Y< X). When a target is no longer within the field of view the node, and therefore no longer identifiable, the node returns to normal operation, reducing frame rate, and notifying the ad-hoc network of the change in status. The other nodes can then increase their frame rate.

The test network performed as designed. Measurements indicate that it is scalable to a large number of nodes and that the performance improves as the number of nodes increases, since there are more radios to share the downlink load.

We conclude that the GNU Radio/USRP platform is capable of fulfilling the downlink communications needs of a small UAV, with data rates up to 1 Mbps in the 300-500 MHz frequency range. Our laboratory prototype demonstrates that cooperative communications techniques are possible with UAV platforms having the computational capacity of a small laptop, using the algorithms developed in this project. Network performance is enhanced.

We recommend the building of a flight proof-of-concept prototype of a GNU Radio/USRP radio for small UAVs implementing the algorithms presented in this report to operate using a low-cost single-board computer like the Texas Instruments *beagleboard*. This is the task of a successor project now under way.

2. Introduction

2.1 Project Objective

The objective of this project is to investigate the application of cognitive radio techniques to coordinated Unmanned Aerial Vehicle (UAV) missions. We intend the project to be the first phase of a three-year effort to develop low-cost cognitive radios for deployment in small UAVs like those described in Section 3.1 *Operational Assumptions* below. The work performed in this first phase addresses two issues:

1. methods by which cognitive radio equipped UAVs working in concert can use cooperative relaying to optimize throughput of high priority information to mission controllers
2. the ability of radios using cognitive techniques to optimize downlink performance under changing channel conditions and evolving mission requirements

Our goal is to explore the potential for the Air Force to replace a UAV's multiple radio platforms with one or more waveform and frequency agile multichannel software defined radios that use cognitive techniques to optimize radio links under changing channel conditions and evolving mission requirements. The project explores methods by which UAVs working in concert can use cooperative relaying to optimize throughput of high priority information to mission controllers.

2.2 Overview of Cognitive Radio, Cooperative Relaying, and Cooperative Communications Literature

2.2.1 Cognitive Radio

For the purposes of this project, a cognitive radio is a frequency and waveform agile software defined radio that is aware of (a) its environment, (b) its user's needs and prerogatives, (c) its own capabilities, and (d) the rules governing its operation and able to take action based on that awareness to accomplish its mission. It is able to learn in the process. We assume that the reader is familiar with the literature of cognitive radio. For a detailed overview, see (Rondeau & Bostian, 2009).

2.2.2 Cooperative Communications

Hong et al., (Hong, Huang, & Chiu, 2007) provide an excellent tutorial on cooperative communications, primarily emphasizing ad-hoc sensor networks. This is based on their paper. They define cooperative communications as taking place in a system where "users share and coordinate their resources to enhance the transmission quality" and point out its close relationship to MIMO systems in exploiting spatial diversity gain. The process normally takes place in two phases: the coordination phase and the cooperative transmission phase. Common techniques include amplify-and-forward (AF), decode-and-forward (DF), and compress-and-forward (CF). The direct signal and the relayed signal are combined at the destination node. The combining process requires strict synchronization between the nodes. The required network operations require close cooperation between distributed nodes and this depends on cross-layer interaction between the PHY and MAC nodes possibly plus higher layers. Many analyses assume full or partial channel state information (CSI) is available to the nodes. Full CSI means that the complex values of the channel coefficients are known (To an RF engineer, these are equivalent to the amplitude and phase of the steady-state transfer function.), and partial CSI means that the magnitudes

of the channel coefficients are known but not their phases. (In RF terminology, this is equivalent to knowing the path loss.) Studies focus on, for example, achieving the lowest outage probability subject to a total transmitted power constraint. Typically amplify-and-forward offers improvement of a few dB over decode-and-forward, at least in theoretical analyses.

Nagaraju et al., (Nagaraju, Ding, Melodia, Batalama, Pados, & Matyjas, 2010) describe a USRP2 implementation of a cross-layer routing and dynamic spectrum allocation algorithm for cognitive radio ad hoc networks. Full details of the algorithm appear in (Ding, Melodia, Batalama, Matyjas, & Medley, 2010). The network was apparently not yet to the testing stage at the time the paper was given. The ad-hoc network described in this paper uses three nodes running in laptop computers and sharing a wired Gigabit Ethernet common control channel. The nodes use USRP2 software defined radio (SDR) platforms as their data channel radios. One node designated as TX functions as a transmitter and two nodes, designated RX1 and RX2, are receivers. The three nodes represent members of a larger ad-hoc network. Data transmission can take place on any one of five frequencies in the 2.4 GHz band, where the ad-hoc network nodes are assumed to be secondary users. (Spectrum sensing to detect activity by primary users was not implemented in the prototype network.) For each data packet to be sent, the transmitter must select the next hop (in this case decide whether to send it to RX1 or RX2), choose the frequency, transmitter power and waveform, and negotiate with the intended receiver to set up and then make the transmission. The process is as follows (quoting from the paper)

> "Every backlogged node *i*, once it senses an idle common control channel performs the following main steps:
>
> 1) For each neighboring candidate next hop, node *i* estimates the maximum achievable link capacity by jointly selecting the transmission channel and power;
> 2) For each neighboring candidate next hop, node *i* calculates the so-called *spectrum utility factor*, defined as the product between the differential backlog of the maximum-differential-backlog session, and the maximum achievable link capacity as estimated in step 1.
> 3) The session corresponding to the optimal spectrum utility factor is scheduled for transmission with corresponding next hop, optimized spectrum and power allocation.
> 4) The signal waveform is selected based on the estimated SINR of the link. In this implementation, we consider BPSK and QPSK as the waveform set. The transmitter compares the expected SINR with a set of pre-defined thresholds to choose the transmission waveform."

To implement the process described above, Nagaraju *et al*., designed their nodes with an internal architecture based on three planes, called the observation, control, and data planes. In their words, "The interaction of the three planes allows allocating the best channel and power for the link with the maximum capacity and session backlog while adaptively regulating the modulation scheme."

2.2.3 Network Coding

Network coding is another form of cooperation that might be considered for the problem at hand. It has distinct origins and a separate literature from that described in (Hong, Huang, & Chiu, 2007). Our

description is based on a summary article by Chou and Wu (Chou & Wu, 2007). Their definition is that "Generally, network coding is the transmission, mixing (or re-encoding) of messages arriving at nodes inside the network, such that the transmitted messages can be unmixed (or decoded) at their final destinations." In a simple example, two data streams arriving at a node could be "mixed by taking their exclusive-OR (XOR) bit-by-bit and sending the mixed stream through the [next] link. This increases the throughput of the network if the two streams can be disentangled before they reach their final destinations. This can be done using side information if it is available downstream." Network coding offers a greater overall throughput for a given network than that available from simple routing alone. Network coding can also minimize the energy required per bit transmitted and minimize the number of hops (and hence the delay) required for a packet to reach its final destination.

2.2.4 Applications to this Project

We have taken an approach to cooperative communications that incorporates elements of the decode and forward (DF) method together with a routing algorithm that configures the radios in a UAV network to give maximum priority and channel capacity to the aircraft with a high priority target. It allows all members of the network to share the downlinking burden. We have not pursued network coding, feeling that the computational requirements are beyond those of the platform we hope to design and fly in a follow on to this project.

2.3 Potential Advantages of Cognitive Radio in UAV Downlinks

It is apparent from measured flight data that the signal almost always fades or drops out whenever the aircraft returns to the same nominal parts of the flight path. We use the term "nominal" because the UAV never returns to exactly the same position with the same attitude and heading. Certain similar combinations of aircraft position vector, aircraft heading vector, and aircraft orientation vector (roll, pitch, and yaw) are associated with known and repeatable signal characteristics. Call the set of these vector components the aircraft data. A brute force approach would be to write a lot of (if, then) statements operating on aircraft data and using them to predict the signal level. This is a huge amount of work. It would have to be hand coded and would be useful only on flight paths for which data were already available. The cognitive approach would be to have the aircraft learn "when I am flying here [above this spot heading in this direction at this altitude] I can expect the signal to drop out if I don't take corrective action in anticipation of the problem. "

The appropriate corrective action depends on the cause of the fade. If it is antenna pattern or terrain shadowing, simply increasing transmitter power or changing modulation characteristics may be sufficient. But if the cause is multipath propagation, a frequency change may be necessary.

3. Methods, Assumptions, and Procedures

3.1 Operational Assumptions

Our work focuses on small UAV platforms flown by the USAFRL. (See Figure 5 of (Kung, Lin, Lin, Tarsa, & Vlah, 2010) for an example.) While the subjects of cognitive radio and cooperative communications both have a rather vast and mathematically complicated literature frequently based on idealized assumptions, we have concentrated on what we think can be achieved in practice using near-term available hardware and software that is compatible with the small UAV platforms of interest.

In this project we have treated the radio nodes (UAVs and earth stations) as primary users of their assigned spectrum. The radios can change frequency as needed to improve performance or avoid interference and jamming, but we have not configured them to operate as secondary users in a dynamic spectrum access (DSA) environment.

This situation is different in several important ways from the ad-hoc sensor network cooperative communications problem as it is commonly treated in the literature.

- Under normal conditions, some or all of the UAVs have their own dedicated downlinks. The network does not route information through multiple UAV nodes unless a downlink fails.
- All of the nodes in the UAV network are in motion. Time synchronization is very difficult or impossible, and the state information for all possible channels varies rapidly with time and aircraft position.

3.2 Use Case: Cooperative Communications for Small UAVs

The primary problem addressed here is this. A coordinated group of UAVs is searching for targets and observing those that it finds. Each UAV has its own downlink to an earth station for transmitting target data and the UAVs are also able to share information over their own ad hoc radio network. The downlinks operate with cognitive radios which attempt to adjust their operating parameters (transmitter power, modulation type and index, operating frequency, data rate, etc.,) as necessary to maintain the acceptable link quality given operational constraints like available battery power. Under ideal conditions each UAV is able to downlink its own target data in real time without needing recourse to the other aircraft. Our first concern is with a situation in which a UAV loses its downlink and particularly one in which that UAV acquires a high value target while the downlink is not available. Our second concern is the related situation where the capacity of the downlink is inadequate for the high-value target data which must be transmitted (in other words, that the downlink latency resulting from a transmission rate limited downlink channel) is too great to allow timely exploitation of a discovered target). In either case a UAV with a high-value target is unable to maintain the required link quality of service on its own downlink and cooperation by the other aircraft is necessary to get target information "home."

3.3 Methods

3.3.1 Partitioning the Problem

Conceptually, our task can be partitioned into a cognitive radio problem (maintaining the downlinks) and a routing problem (how to route a data stream from a UAV which has a target but no downlink to a UAV which has a downlink but, ideally, no target). The routing problem also opens the door to network coding, which is a generalization of routing (Chou & Wu, 2007). We will address the cognitive radio problem first, describing our effort to simulate cognitive radio behavior in real UAV links.

3.3.2 Simulating Cognitive Radio Behavior in Real UAV Downlinks

3.3.2.1 Simulator Design

Partitioning the problem into a radio problem and a routing problem, we investigated the capability of cognitive radios based on the GNU Radio/Universal Software Radio Peripheral (USRP) open-source platform to meet the needs of the UAVs of interest. GNU Radio is an open source software system for building software defined radios. See (Welcome to GNU Radio, 2011). USRP is a generic term for a family of low cost software defined radio (SDR) platforms marketed by Ettus Research (Ettus, 2011). GNU Radio code usually runs in conjunction with USRPs. Both are widely used, and we assume that the reader is generally familiar with them. For a comprehensive overview, see (Rondeau & Bostian, 2009).

Figure 1 illustrates the simulation system developed for this project. The block labeled "channel simulation" is capable of inserting time-varying values of transmitter and receiver antenna gain, path loss, noise, and multipath into the downlink. For simplicity of operation, we do the insertion at complex baseband in GNU Radio code, rather than at RF between the transmitter and receiver. So long as the link is linear, the two methods (baseband and RF) are equivalent.

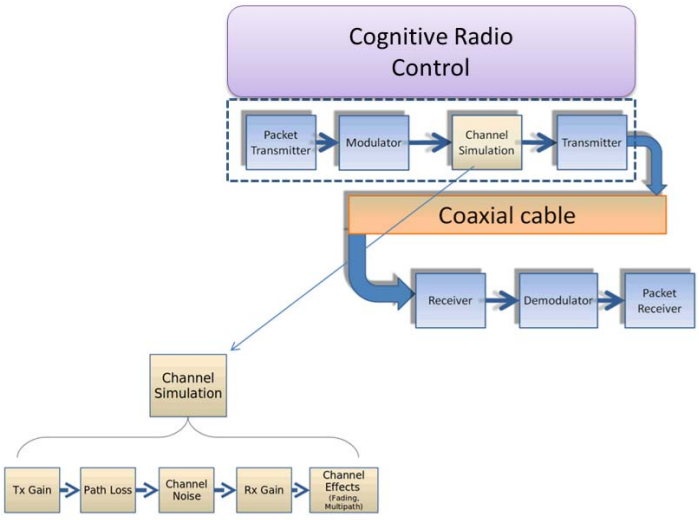

Figure 1 GNU Radio/USRP-Based UAV Downlink Simulator

3.3.2.2 Radio Platform Tests

We did a number of experiments to investigate the behavior and reliability of data links between USRPs. A series of tests was conducted transmitting data between two USRPs over coaxial cable to eliminate any negative external effects due to the over the air (OTA) environment. Seven tests were conducted. 10,000 digital packets of length 1500 bytes were transmitted in a continuous stream at a bit rate of 125kbps and modulated using differential binary phase shift keying (DBPSK). The test configuration is shown in Figure 2. A transmitting node, consisting of a general purpose processor (GPP) based computer and USRP for the RF front end, was connected to a receiving node–also consisting of a GPP based computer and RF front end, using RG 180 coaxial cable. RG 180 attenuates at approximately 12 dB per 100 feet at 400 MHz. Less than 10 feet of coaxial cable was used to connect the transmitter and receiver during the tests; at this distance, the attenuation is almost negligible (>1.2 dB) within the scope of the tests. A spectrum analyzer was connected inline to allow monitoring of power levels. As the receiving node received packets from the transmitter, information about the number of packets received and the number of packets that passed a data integrity check was written to a file. This information was later used to determine the performance of the USRPs under certain conditions. Results appear in Section 4.1 Radio Platforms.

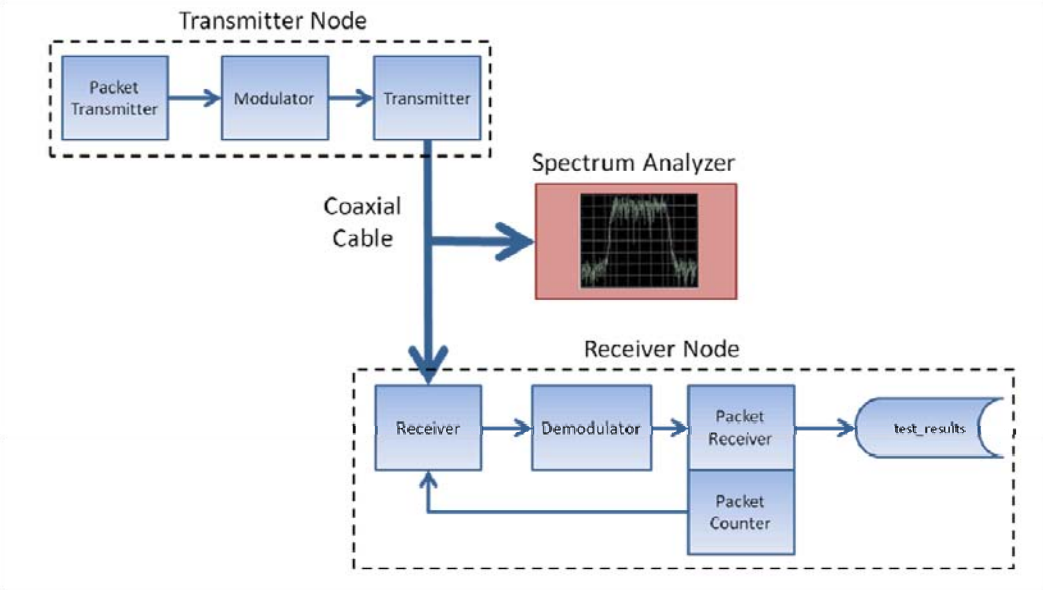

Figure 2 Block diagram representing transmitter and receiver setup and node interconnect configuration for USRP and USRP2 performance tests.

3.3.2.3 Incorporating Measured Data

The experimental program described in (Kung, Lin, Lin, Tarsa, & Vlah, 2010) and referenced above provides measured RSSI values taken at 0.2 second intervals from representative UAV links to earth stations on vehicles, towers, etc. Accompanying these are data on link quality (number of bad packets, number of good packets, throughput, etc.) and aircraft position, heading, attitude, etc. From the RSSI values it is easy to extract channel fade values (in dB with respect to the observed peak RSSI value) and to estimate the observed signal-to-noise ratio as a function of time.

Received signal strength (RSSI) from the UAV is determined by free space path loss, multipath propagation, and the interaction of the transmitting and receiving antenna patterns (shadowing by the airframe and terrain is inseparable from the antenna patterns). A problem in simulating link behavior is that an apparent fade caused by antenna and terrain effects does not have the same effect on link quality as a dB-equivalent fade caused by multipath effects. The antenna/terrain fade can easily be overcome by increasing the transmitter power, and its effects are easy to calculate, while a multipath fade may well be accompanied by dispersion which cannot be corrected by adaptive control and whose effects on link quality cannot easily be computed.

Due to delays in gaining access to the USAFRL data and in receiving the software necessary to manipulate it, we were unable to incorporate measured data into the simulator of Figure 1. (We received the software on December 14, 2010, nine days before the project ended.) We plan to do that in our subsequent project when we will test a flight version of a cognitive radio for small UAVs.

3.3.3 The VT Test Network

3.3.3.1 Design

The U.S. Air Force Research Laboratory has sponsored a comprehensive survey of platforms and test beds for experimental evaluation of ad-hoc networks (Chowdhury & Melodia, 2010). That reference provides a comprehensive description of all the hardware and software options currently available to cognitive radio researchers, including the GNU Radio/USRP platforms used in our work.

Consistent with the focus of this project, we designed a test network emphasizing cognitive radio solutions to the UAV problem, rather than solutions based primarily on routing. Thus we postulate a network consisting of five nodes representing UAVs (We call these “UAV nodes”) and a node representing an earth stations (“E/S node”). The uplinks and downlinks all operate in the 450 MHz FRS band, 462.5625 to 467.7125 Mhz, whose characteristics are very similar to those of the military 310-390 MHz band.

The UAV nodes organize themselves into an ad-hoc network using WiFi. This network both performs the same coordination role as the wired Ethernet common control channel in (Nagaraju, Ding, Melodia, Batalama, Pados, & Matyjas, 2010) and also provides a path over which the UAV nodes can pass traffic among themselves. While we could have implemented the ad-hoc network using USRPs, we saw no need to re-invent the capabilities of WiFi, already available at very low cost in very small, power-efficient packages. Commercial modems are also available that, like WiFi cards, provide a TCP/IP interface to the rest of the network.

The UAV nodes’ uplinks and downlinks use cognitive radios, running on USRP/GNU Radio platforms. As we build up their capabilities, these radios will be able to take intelligent action to compensate for changes in link quality resulting from the aircraft position and orientation and from propagation effects like multipath and shadowing. We will be able to replace the simple over-the-air radio channels in our laboratory with simulated channels that play back UAV propagation data recorded collected by AFRL and discussed in detail in a later section of this report.

3.3.3.2 Operation

When the WiFi ad-hoc network forms, one of the UAV nodes is designated the “edge “node. It establishes a link with an E/S node and hosts the downlink. All of the downlink-capable UAV nodes take turns serving as edge node. In our initial network design the time interval that any one node serves is controlled by a timer. When the timer expires, the current edge hands off the downlink to a new node, and the new node become the edge until the next expiration of the timer.

In order to demonstrate that the intelligence in the cognitive radio can also be used as the brain of the UAV, the UAV nodes perform their missions as follows. Every node in the UAV network takes pictures with its attached camera at a rate of X frames per second (FPS). These frames are forwarded over the ad-hoc network to the edge. The edge then transmits the frames to the E/S, which forwards it to a central location for display. If a UAV node identifies a target in a particular frame it has captured, it 1) notifies all the nodes in the network that it has acquired a target, and 2) increases its capture rate to 10X. Upon receiving notification that another node has identified a target, a node reduces its own capture rate to Y (Y< X). When a target is no longer within the field of view the node, and therefore no longer identifiable, the node returns to normal operation, reducing frame rate, and notifying the ad-hoc network of the change in status. The other nodes can then increase their frame rate.

3.3.3.3 Implementation Details

3.3.3.3.1 Overview of the Network

The lab implementation consists of one to four laptops serving as UAV nodes, one laptop serving as the earth station, and one desktop serving as the headquarters/image display machine. The UAV nodes are all part of an ad-hoc WiFi-based network using standard WiFi hardware. Each UAV node is also connected to an USRP RF front end which— in association with GNU Radio—provides non-WiFi RF communications. See Figure 3.

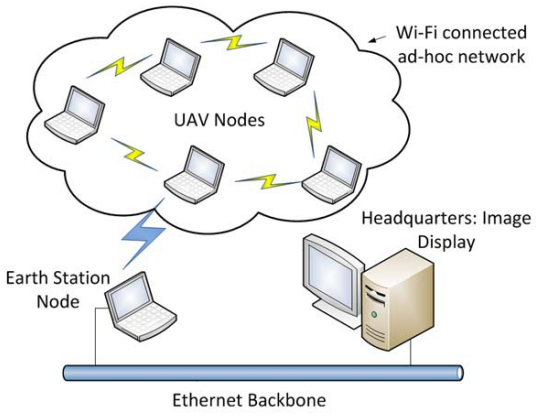

Figure 3 CWT AFRL UAV network Implementation.

Upon startup, the UAV nodes establish and join and ad-hoc network. One of the nodes is designated the "edge" node, and is responsible for providing an RF backbone downlink to the earth station node. The edge node hosts the downlink for a limited period of time before passing off responsibility for hosting the downlink to another UAV node. This new node is now designated the edge, and all other UAV nodes are informed accordingly.

Every non-edge node in the UAV network takes pictures with an attached webcam at a standard frame rate. The captured frames are forwarded over the network to the edge, and stored in a local database. The edge removes the pictures from the database and forwards them over an RF link to the earth station node. The images can be displayed locally at the earth station, but are currently forwarded to the headquarters node for display (Figure 4).

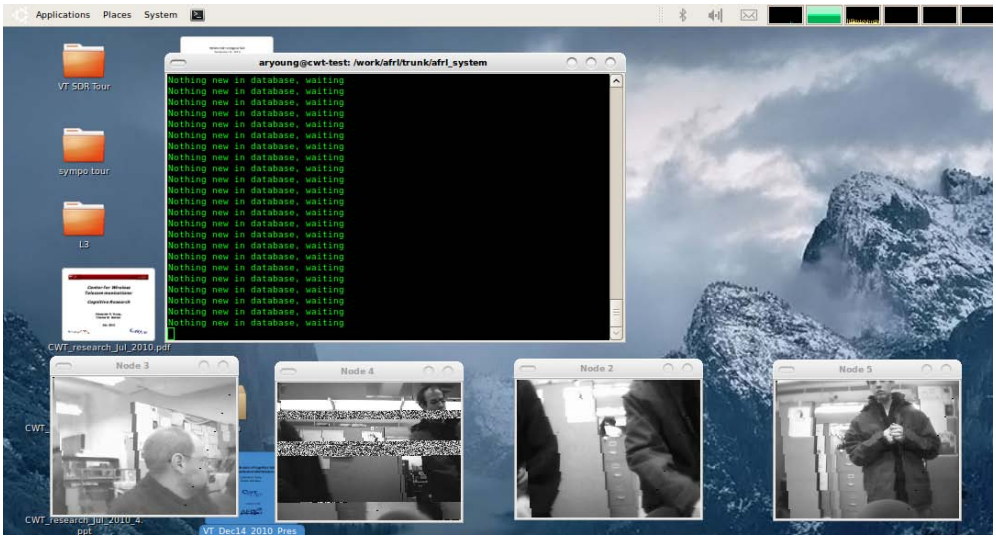

Figure 4 Captured frames displayed on screen of headquarters system.

Using computer vision and target identification algorithms, the UAV nodes are able to determine if any frame contains the image of a target of interest. If a UAV node identifies, or acquires, a target, it 1) notifies its peer nodes that it has acquired a target, and 2) increases its capture rate. Upon receiving notification that another node has acquired a target, a node reduces its own capture rate. When a UAV node loses a target that it has previously acquired, it returns to normal operation, reducing frame rate and notifying the network of its change in status. In this way, the UAV network dynamically changes its configuration to adapt to changing scenarios.

3.3.3.3 2 UAV Node

UAV nodes use a threaded architecture with two processes running in parallel (Figure 5). The primary UAV controller spawns a socket server at startup. The socket server handles incoming TCP/IP socket connections from other UAV nodes, while the primary thread is responsible for managing the remaining UAV subsystems: vision module, RF module, message handler, and knowledge base (Figure 6). The

vision module controls the UAV's visual sensor, in this case a webcam, and uses computer vision algorithms to identify targets. The RF module controls the non-WiFi RF communication component, using GNU Radio software and USRP RF front end to maintain a downlink with the earth station node. The message handler handles outgoing messages in the ad-hoc network, using TCP/IP sockets to communicate with other UAV nodes. The knowledge base is a central repository for all system data and configuration information. Each UAV node subsystem has direct access to the knowledge base, allowing the different components and modules within a single UAV to communicate and coordinate their actions. The UAV node subsystems are discussed in more detail below.

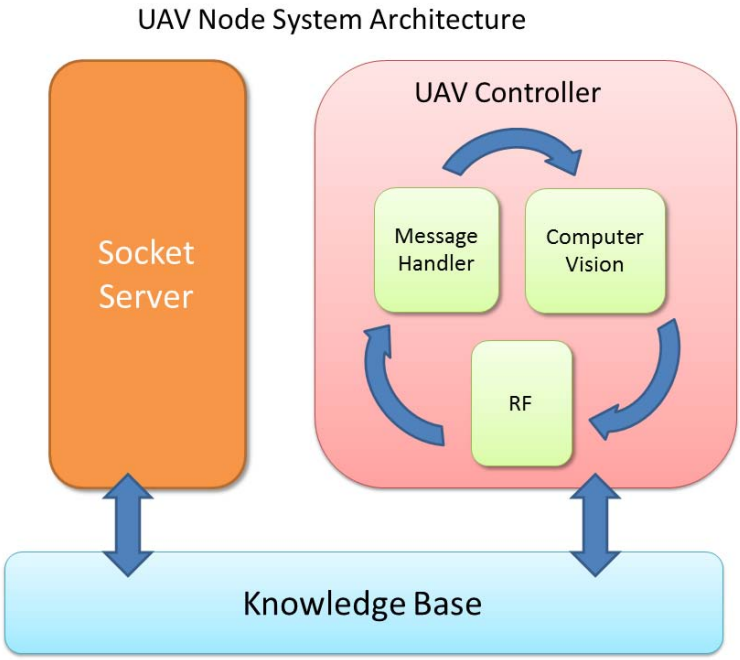

Figure 5 UAV node system architecture diagram.

3.3.3.3.3 UAV Controller

The UAV controller shown in Figure 6 is the heart of the UAV system. The controller initializes all the components, checks for ad-hoc network connectivity, and then starts the socket server in a new thread. This allows the socket server to run independently of the rest of the modules, responding to incoming communications requests on demand. The controller is responsible for managing the remaining UAV subsystems: vision module, RF module, message handler. The knowledge base is directly accessible by the controller and the socket server, as well as the controller subsystems.

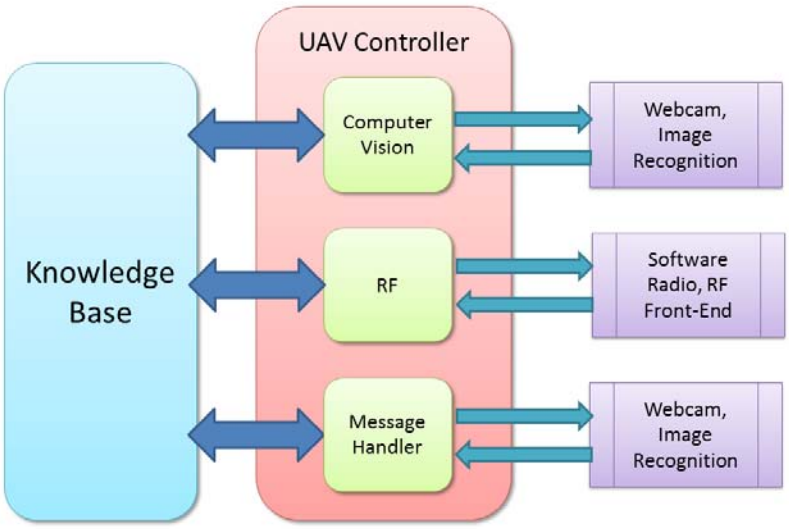

Figure 6 UAV controller architecture.

The controller's main functionality exists in its *run()* function. The *run()* function contains a loop that calls either the vision subsystem or the RF subsystem, depending on the UAVs current mode, then calls the message handler. The RF subsystem is used when the UAV node is functioning as the edge node, and the vision subsystem is used when the UAV node is not functioning as an edge node. The *run()* loop repeats continuously during UAV node operation. See Figure 7.

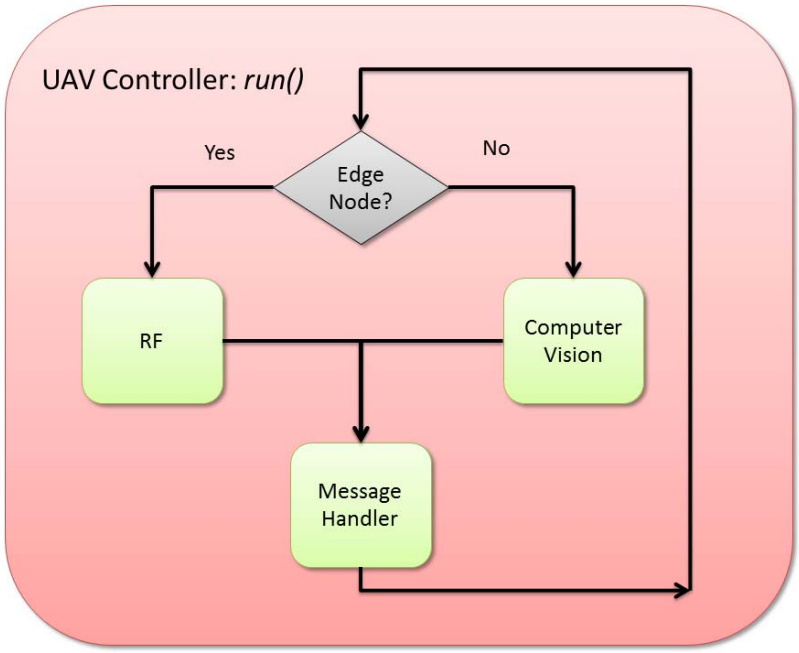

Figure 7 UAV controller primary loop flow.

In order to provide resource conservation across the entire UAV network, the UAVs share responsibility for hosting the downlink. Each node hosts the down link for approximately a minute (the actual duration is determined by number of iterations through the main run() loop. When the run() loop starts, the current edge node sends a message to the other nodes requesting information on who (i.e., which node) can next host the link. After the edge node receives replies from other nodes in the network, it appoints the next host in round-robin fashion, based on the numerical order of the IP addresses of the nodes in the ad-hoc network. The entire network is notified of the new edge/downlink host. The new edge takes up this role immediately, and the rest of the nodes in the network now forward their images to the new edge. This handoff occurs regularly, ensuring that no node spends too long at any one time hosting the high power RF downlink.

3.3.3.3.4 Socket Server

The socket server, in conjunction with the message handler, provides inter-node communication for UAV nodes. The socket server is responsible for managing incoming socket connections from other UAV nodes. Upon startup, the socket server opens a blocking TCP/IP socket on port 42424 and listens for incoming socket connection. When a UAV node connects to the socket server, it is for the purpose of requesting or providing information. When the UAV node disconnects from the socket server, the connection is closed on both sides. At this time, the socket server has received a message. The socket server processes the incoming message, and reopens the socket, listening for new connections.

Possible messages that the socket server can receive are: EDGE_QUERY, EDGE_QUERY_RESPONSE, TARGET_ACQUIRED, TARGET_LOST, and NEXT_EDGE. Each of these messages has a different meaning.

An EDGE_QUERY message is sent by the current edge to every other node. The message indicates that the current edge wants to know who is capable of hosting the downlink. When the socket receiver receives this message, it sets the *send_edge_query_response* flag in the knowledge base.

An EDGE_QUERY_RESPONSE is sent in response to an EDGE_QUERY. The message EDGE_QUERY_RESPONSE is followed by an IP address, and either *True* or *False*, indicating whether the node sending the EDGE_QUERY_RESPONSE and identifying itself by its IP address can host the RF downlink. The socket server stores this information in the knowledge base, in *capable_edges*, a list of Boolean values referenced against a list of all IP addresses in the network, stored in *full_network*. The socket server also keeps track of the number of EDGE_QUERY_RESPONSEs received, storing this value in *edge_query_response_count*.

A TARGET_AQCUIRED message is sent when a UAV node identifies a target using its computer vision subsystem. When the socket server receives this message, it immediately sets the *else_target_acqd* flag in the knowledge base.

A TARGET_LOST message is sent when a UAV node is no longer able to identify a target using its computer vision subsystem. When the socket server receives this message, it resets the *else_target_acqd* flag in the knowledge base.

A NEXT_EDGE message is sent by the current edge to all the nodes indicating which node is next responsible for hosting the RF downlink. A NEXT_EDGE message is followed by the IP address of the node to which the downlink will pass. A socket server that receives this message checks the IP address against the knowledge base value *self_address*. If the two values match, the socket server sets the flag *self_edge*; otherwise it resets the flag.

3.3.3.3.5 Vision Module

Active only for UAV nodes that are not currently edge nodes, the UAV node's vision subsystem is responsible for taking pictures, analyzing the using image identification algorithms, and sending images to the appropriate edge node for transmission to the Earth Station node.

The core of the vision module is its sensor, in this case a standard webcam. The vision module uses the webcam to take a picture, which it then analyzes for image identification. The OpenCV computer vision library provides vision capabilities and image identification algorithms. Once the vision module has taken a picture, it uses OpenCV to analyze the picture for the purpose of target identification. The target in this case is a checkerboard pattern. Figure 8 shows the vision module identifying the target in a picture it has taken.

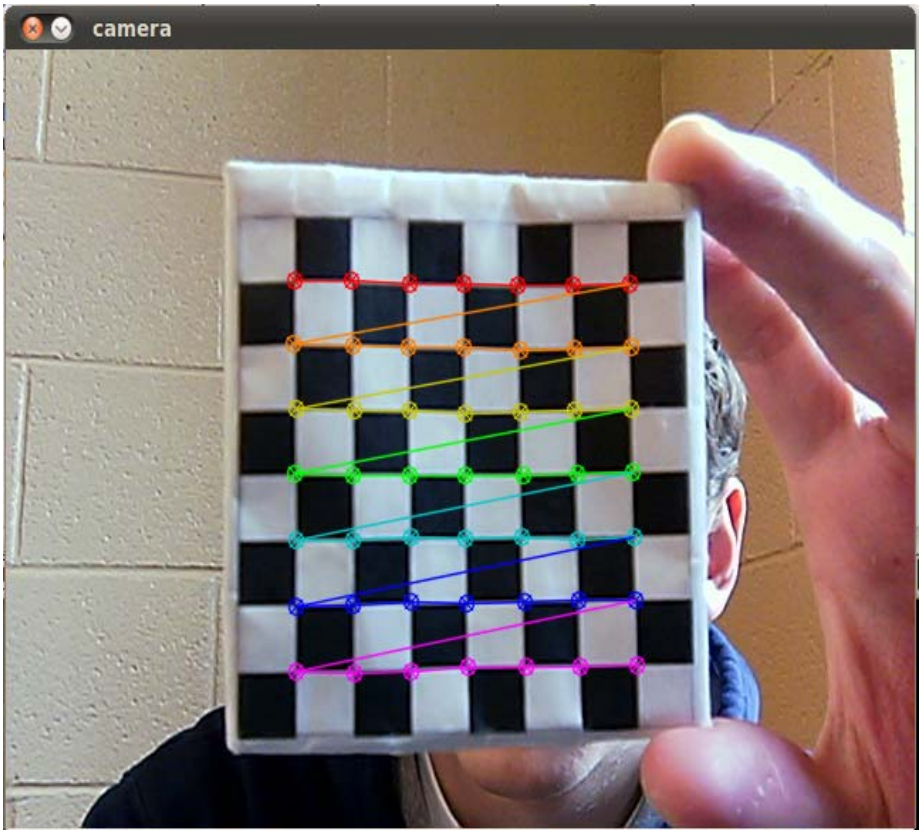

Figure 8 Screen capture showing image identification using OpenCV library

Picture rate: the computer vision module is capable of adjusting its picture taking rate, adapting to network load/traffic priority. In the absence of any external information, the vision module takes pictures at a base rate. If the UAV has been notified that another node has identified a target, this information is available in the knowledge base. If flag else_target_acqd is set, the vision module reduces its frame rate to half the base rate. Alternatively, if the vision module detects a target in a particular frame, it sets flag self_target_acqd and increases its frame rate to twice the base rate.

In addition to grabbing image frames and processing them using image identification algorithms, the vision module also forwards the images directly to the edge node for retransmission over the down link. The vision module uses a set of database utilities to access directly the database on the edge node, and deposits the image, along with information about the image size and its node ID (the last octet of its IP address) into the database. This is possible because the edge node uses a database server to handle all its database transactions. See the flowchart in Figure 9.

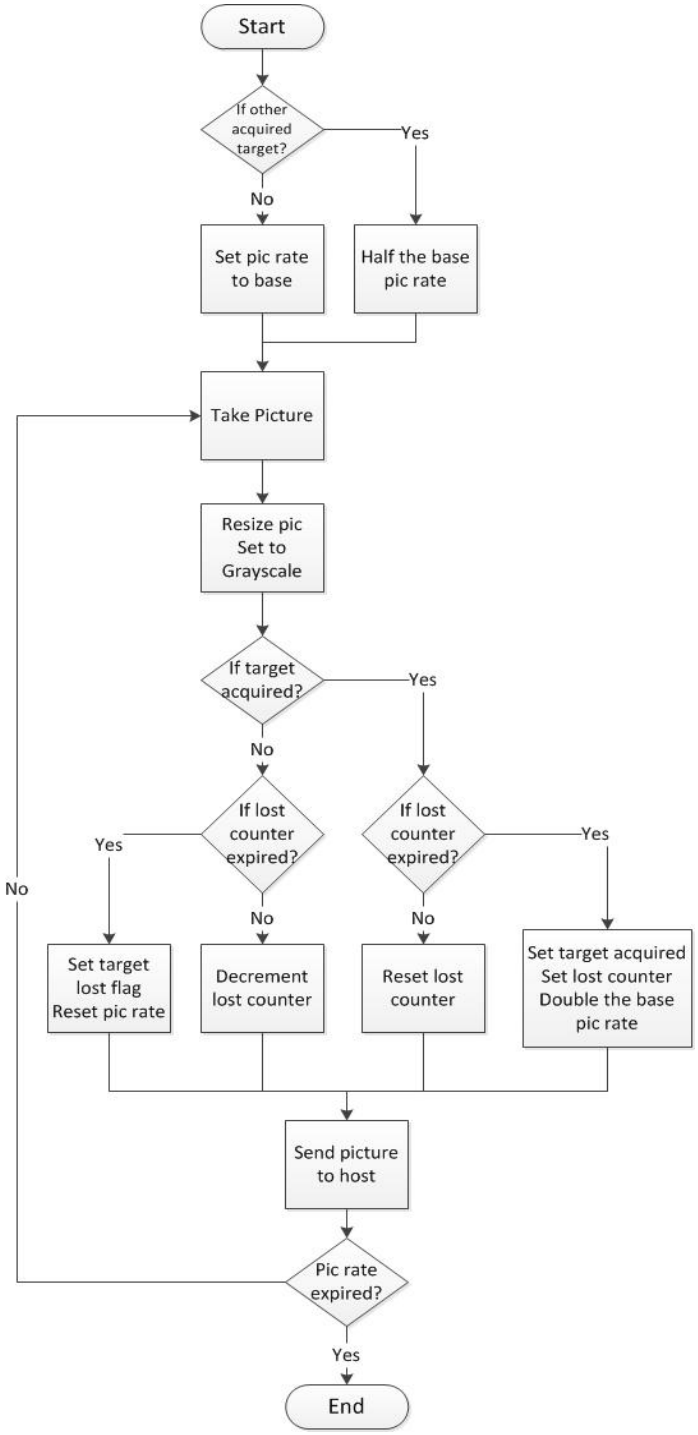

Figure 9 Diagram showing program logic and flow for vision subsystem.

3.3.3.3.6 RF Module

The RF module provides the core functionality for edge nodes. The RF module is responsible for supplying the downlink that the rest of the network uses to relay pictures to headquarters. Reconfigurable RF communications are provided by the software radio toolkit GNU Radio, and Ettus Research Universal Software Radio Peripheral (USRP) RF front end. GNU Radio provides the GPP based software for the radio DSP components, such as waveform generation, sampling, and filtering. The software radio uses a flow graph structure to provide radio capabilities. The RF module is implemented using a three layer stack design. See Figure 10.

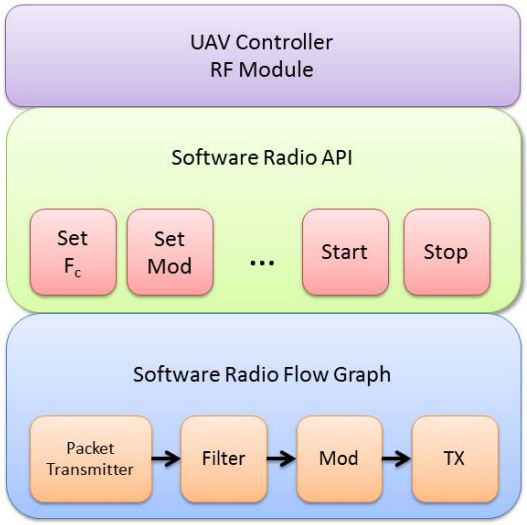

Figure 10 The RF module's 3-level stack design.

The UAV controller operates at the top level, implementing and operating the radio through the API (at the middle layer).

The middle layer of the RF stack is the software radio API, and enables rapid radio reconfiguration. The API provides several methods for configuring aspects of the radio flow graph, including setting center frequency, modulation, and bitrate; building the radio flow graph before use and deleting it after use; functions to start and stop the radio; and convenience functions to handle several tasks at once, such as building, configuring, and starting the radio all at once. The API is capable of configuring and controlling an RF energy sensor, but the UAV controller does not currently use an RF sensor. This will be implemented in future work.

The bottom layer of the stack is the radio flow graph itself. The flow graph is made up of individual radio-related and general DSP blocks. The API configures each of these blocks using the necessary parameters and connects the blocks together to create a radio flow graph. Data packets pass through each block in succession before finally being transmitted over the air.

While the vision module handles image processing for the UAV node, the RF module provides the gateway functionality for image transport between the UAV ad-hoc network and the ground network

represented by the earth station and headquarters nodes. Within the API layer exists logic for removing image frames from the edge node's image database and preparing the images for transmission over the RF link.

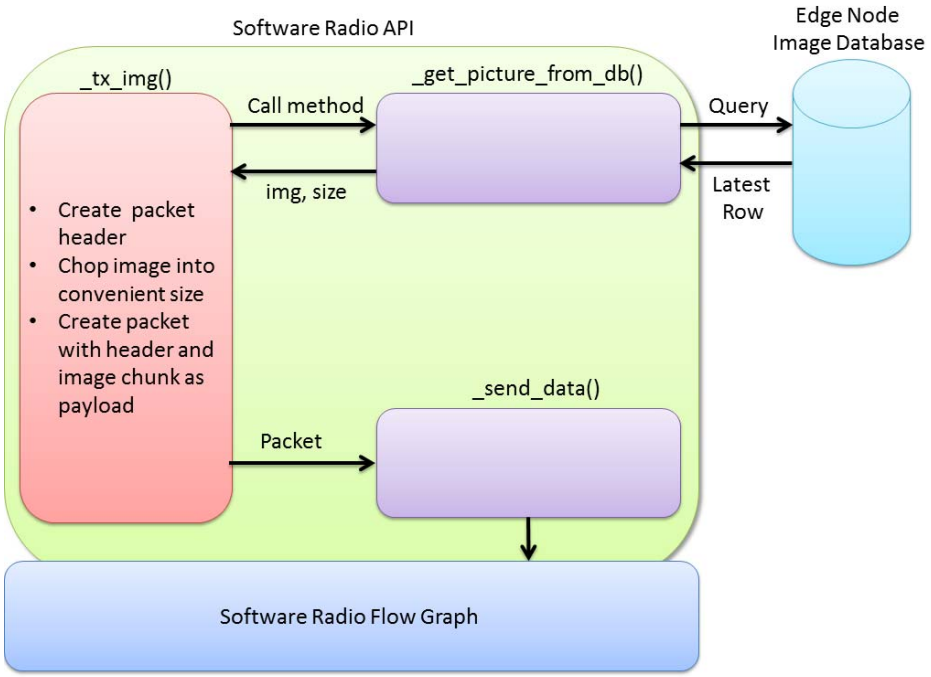

Figure 11 Preparing image frames for transport over RF downlink.

Figure 11 above summarizes the steps taken to prepare the image frames for transmission to the earth station node using the RF downlink. The private method _tx_img() coordinates the actions taken to process and prepare the image for transmission. _tx_img() calls the private method _get_picture() to get the image from the edge node's local image database, where all the images from the UAV network are stored. _get_picture() provides a query to the database requesting the most recent image in the database. The database returns the most recent image, along with data associated with that image, namely the image's size in pixels (height x width). _get_picture() reformats the database query results into a useful format: the image is converted to a string, and the size is converted to a tuple of strings. The tuple and string are then returned to _tx_img(). _tx_img() is responsible for creating the data packet that will be transmitted. Using the size tuple, _tx_img() creates a packet header consisting of packet number, image size, and end-of-image (EOI) flag. _tx_img() then chops up the image string into manageable pieces, and creates a packet with the header and image string chunk as payload. The completed packet (Figure 12) is sent to the software radio flow graph by _send_data(), and the flow graph completes the process by transmitting the packet over the air. _tx_img() continues to create

headers and new packets until it has completely packetized and relayed an entire image frame. _tx_img() then calls _get_picture() once more to obtain a new image frame for processing and relay. This process continues as long as a UAV node is functioning as the edge node.

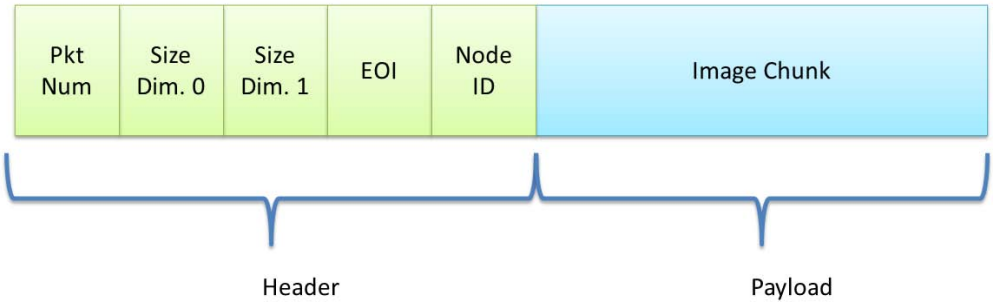

Figure 12 Packet structure for data used in RF link.

3.3.3.3.7 Message Handler

The message handler module is responsible for all outgoing communication from the uav node to its peers over the adhoc uav wifi network. While the UAV is operating, each of the components read and record information from the knowledge base. When the message handler runs, it reads values from the knowledge base and sends messages to other nodes as required.

Messages are simple text strings, and are sent in one of two ways: with direct one-on- one communication, or in broadcast. The message handler's send() method handles on communication by opening a socket connection to a single host and sending a message. The broadcast() method operates in a similar manner, but repeats the send process for every node in the network.

The message handler also contains the private method _find_usrp. _find_usrp is used to determine if a UAV node is capable of hosting the downlink and becoming an edge node by determining if the UAV is currently connected to a USRP, the RF front end required for downlink communications.

The message handler's run() function is a simple finite state machine with five states, responsible for determining the required communications. The finite state machine runs through each of the five states in turn, determining if action must be taken.

If in the knowledge base the self_target_acqd flag is set and the target_acqd_message_sent flag is not set, then the message handler broadcasts TARGET_ACQUIRED to all nodes, then sets the target_acqd_message_sent flag.

If in the knowledge base the self_target_acqd flag is not set and the target_acqd_message_sent flag is, indicating that the UAV vision subsystem has lost contact with a target, then the message handler broadcasts TARGET_LOST to all nodes, then resets the target_acqd_message_sent flag.

If the knowledge base flag send_edge_query is set, the message handler boradcasts the message EDGE_QUERY and resets the send_edge_query flag.

If the send_edge_query_response flag is set, the message handler determines if the UAV node is capable of hosting the downlink as an edge node by calling _find_usrp. This returns a Boolean value (True or False) corresponding to the UAVs edge node capability. The message handler then constructs a message containing the UAVs own network address and the Boolean value, and sends the message to the current edge node.

Finally if the value for next_edge in the knowledge base is not “None”, the message handler broadcasts a message containing NEXT_EDGE followed by the value of next edge, and sets the value of next_edge to none.

3.3.3.3.8 Knowledge Base

The knowledge base is a system module implemented in Python and is the central storage location for all system data, flags, and configuration information. Every component and subcomponent has direct access to the knowledge base for reading and writing.

To ensure that all components can access the knowledge base without interfering with other components, knowledge base access is associated with a thread lock. The thread lock acts like a token. A component can only write to the knowledge base if it is in possession of the lock, and therefore must request the lock before any knowledge base write operation. A single thread lock ensures that only one component is writing to knowledge base at any time.

Knowledge base read access is generally done through the use of the get_state() method. get_state() provides a dictionary list of all variables in the knowledge base and their values at the time the function is called. Occasionally a component may need more up to date information on a single variable. In this case, the component requests the thread lock as above. This ensures that the component has the most up-to-date value; no other component can update the knowledge base without the thread lock.

Table 1 System flags in knowledge base.

Flag	Flag set	Flag not set
self_target_acqd	• UAV’s local vision subsystem has located a target • Local UAV increases image capture rate	• If target_acquired_msg_sent set, local UAV has lost target • Send TARGET_LOST message • Reset target_acquired_msg_sent
target_acquired_msg_sent	• UAV network has been notified that UAV’s local vision subsystem has identified a target	• No action required
else_target_acqd	• Another UAV in network has identified a target • Local UAV decreases image capture rate	• Another UAV has lost its target • Local image capture rate

		returns to normal rate
self_edge	• Local UAV is currently the edge • Local UAV uses RF subsystem instead of vision subsystem	• Local UAV is not the edge • Local UAV uses vision subsystem
send_edge_query	• Send EDGE_QUERY message to other UAVs in network	• No action required
send_edge_query_response	• Send EDGE_QUERY_RESPONSE message ({True|False}) to edge node indicating if local UAV can host the downlink	• No action required

Table 2 Data entries in knowledge base.

Entry	Description	Parameter
edge_query_response_count	Running count of nodes that have responded to edge_query message.	Positive integer, maximum value is number of nodes in network -1
full_network	IP address of all nodes in UAV network	List of strings, each string is an IP address in dotted decimal form
capable_edges	UAV nodes capable of becoming next edge	List of Boolean values, in order corresponding to full_network
port	TCP/IP port number socket server listens on	Integer, any valid unused TCP/IP port. Default is 42420
self_address	UAV's own IP Address	String containing IP address in dotted decimal form
curr_edge	IP address of current UAV edge node	String containing IP address in dotted decimal form
next_edge	IP address of next UAV edge node	If next edge has been indicated, string containing IP address in dotted decimal form, otherwise None

3.3.3.9 Earth Station

The earth station node is the other end of the RF link hosted by the UAV edge node. The earth station receives every packet that comes in from the edge node, reassembling the chopped up images as it receives every chunk. It does this in a process that mirrors the process used to send the packets. As the image chunks are received, they are concatenated into a larger and larger string until the earth station receives a packet where the EOI flag is set (binary 11111111). This indicates that the last chunk of a single image has been sent. Using information stored in the packet header, the earth station reconstructs the image, then forwards the image to the headquarters node. Directly interfacing with the database on the headquarters node, the earth station adds an entry into the database that contains the image, the image size in pixels, along with a node ID, identifying the UAV node that originally took the image.

3.3.3.10 Headquarters

The headquarters node is the final destination of all data gathered by the UAV nodes; in this case images captured with webcams. The headquarters node displays all the images it receives for easy viewing (Figure 13). As the earth station node adds pictures to the headquarters database, the headquarters node removes the images from the database, converts them from an image string to actual image, and displays each image in a window, depending on which UAV node captured the original image. Figure 13 shows the headquarters node displaying images received from two different UAV nodes in separate windows.

Figure 13 Close up view of headquarters display showing images captured by two different UAV nodes displayed in separate windows.

4. Results and Discussion

4.1 Radio Platforms

Using the setup of Figure 2 we tested several combinations of Ettus Research USRP boards. For transmission between two USRP2s, each using an Ettus research 400 MHz daughter board, and centered at 450 MHz, 100% of the packets were received, and 94.8% of the received packets were without error. For transmission between two USRPs, using the same two Ettus research 400 MHz daughter boards and again centered at 450 MHz, 100% of the packets were received and without error.

Tests conducted at higher frequencies (above 1.5 GHz) had much poorer results. For this reason, we concentrated our efforts in the 400 MHz band. This is consistent with the Air Force 310-390 MHz military band measurement program.

Table 3 shows typical test results. From it we conclude that cognitive radios based on the GNU Radio/USRP platform can support data rates of at least 1 Mbps for small UAV platforms using DBPSK, DQPSK, and GMSK modulations.

Table 3 Experimental results using USRP 400 MHz daughter board.

Target rate	Actual bit rates (bits per second)				
(bits per second)	DBPSK	DQPSK	D8PSK	GMSK	
60k	60.096k			60.096k	
75k	75.075k			75.075k	
80k	80.128k			80.128k	
100k	100k			100k	
110k	109.649k	111.606k[1]		109.649k/109.89k[3]	
115k	115.207k	115.206k		114.678k/114.942k[3]	
120k	120.192k	[2]		120.192k	
125k	125k	125.312k		125k	
130k	130.208k	229.87k		129.87k/130.208k[3]	
135k	135.135k	134.77k		135.135k	
145k	145.348k	144.926k		145.348k/144.927k[3]	
150k	150.602k	150.15k		150.602k	
175k	175.056k	175.438k		176.056k/175.438k	
200k	200k	200k	199.998k	200k	
225k	225.225k	225.224k		225.225k	
250k	250k	250k		250k	
300k	297.619k	301.204k	300k	297.619k/301.204k[3]	
350k	347.222k	352.112k	350.466k	347.222k/352.112k[3]	
400k	396.825k	400k		396.825k	
450k	446.428k	450.45k		446.428k/480.45k[3]	
500k	500k	500k		500k	
750k	757.575k	757.57k		757.575k	
1000k	1000k	1000k		1000k	
	notes:	1. Minimum Rx bitrate			
		2. No mutually achievable rate between Tx/Rx			
		3. Actual Tx/Rx rates as reported by GNU Radio			

4.2 Network Simulations

Using the network described in Section 3.3.3 *The VT Test Network*, Time stamps were inserted into the system each time an image was received at the earth station. The entire system was then run for approximately an hour (500 controller iterations), and the time stamps were recorded throughout. The system was run initially using 2 UAV nodes. The test was repeated for 3, 4, 5, and 6 nodes. The results are summarized in **Error! Reference source not found.**. The minimum sample size was 1088 data points, for the 2 node test. The columns show the maximum, minimum, mean, and standard deviation of the time intervals between successive receptions of an image at the earth station. It is clear that the UAV network scales well with additional nodes. The network test setup is shown in Figure 14 and Figure 15.

Table 4 Summary of system testing.

Test	t_{max} (sec)	t_{min} (sec)	t_{mean} (sec)	t_{σ} (sec)
2 Nodes	12.390	0.546	1.943	2.253
3 Nodes	10.738	0.319	0.952	1.162
4 Nodes	10.315	0.546	0.627	0.515
5 Nodes	10.225	0.509	0.618	0.495
6 Nodes	7.099	0.546	0.584	0.192

Figure 14 UAV network running tests with 6 nodes. This image shows the 6 nodes in operation.

Figure 15 UAV network running tests with 6 nodes. This image shows the USRP RF front ends used.

5. Conclusions

The GNU Radio/USRP platform is capable of fulfilling the downlink communications needs of a small UAV, with data rates up to 1 Mbps in the 300-500 MHz frequency range. Our laboratory prototype demonstrates that cooperative communications techniques are possible with UAV platforms having the computational capacity of a small laptop, using the algorithms developed in this project. Network performance is enhanced.

6. Recommendations

We recommend the building of a flight proof-of-concept prototype of a GNU Radio/USRP radio for small UAVs implementing the algorithms presented in this report to operate using a low-cost single-board computer like the Texas Instruments *beagleboard* (beagleboard.org, 2011). This is the task of a successor project now under way.

7. References

beagleboard.org. (2011, March 22). Retrieved March 22, 2011, from beagleboard.org: http://beagleboard.org/

Welcome to GNU Radio. (2011, March 22). Retrieved March 22, 2011, from GNU Radio: http://gnuradio.org/redmine/wiki/gnuradio

Chou, P. A., & Wu, Y. (2007, September). Network Coding for the Internet and Wireless Networks. *IEEE Signal Processing Magazine*, pp. 77-85.

Chowdhury, K. R., & Melodia, T. (2010, September). Platforms and Testbeds for Experimental Evaluation of Cognitive Ad Hoc Networks. *IEEE Communications Magazine*, pp. 96-104.

Ding, L., Melodia, T., Batalama, S. N., Matyjas, J. D., & Medley, M. J. (2010). Cross-Layer Routing and Dynamic Spectrum Allocation in Cognitive Radio Ad Hoc Networks. *IEEE Transactions on Vehicular Technology, Vol. 59, No. 4*, 1969-1979.

Ettus, M. (2011, March 22). *Welcome to Ettus Research*. Retrieved March 22, 2011, from Ettus Research: http://www.ettus.com/

Hong, Y.-W., Huang, W.-J., & Chiu, F.-H. K.-C. (2007, May). Cooperative Communications in Resource-Constrained Wireless Networks. *IEEE Signal Processing Magazine*, pp. 47-57.

Kung, H., Lin, C.-K., Lin, T.-H., Tarsa, S. J., & Vlah, D. (2010). A Location-Dependent Runs-and-Gaps Model for Predicting TCP Performance Over a UAV Wireless Channel. *MILCOM* (pp. 1058-1066). San Jose, CA: IEEE.

Nagaraju, P. B., Ding, L., Melodia, T., Batalama, S. N., Pados, D. A., & Matyjas, J. D. (2010). Implementation of a Distributed Joint Routing and Dynamic Spectrum Allocation Algorithm on USRP2 Radios. *Sensor Mesh and Ad Hoc Communications and Networks (SECON), 2010 7th Annual IEEE Communications Society Conference on* (pp. 1-2). Boston, MA: IEEE Communications Society.

Rondeau, T., & Bostian, C. (2009). *Artificial Intelligence in Wireless Communications.* Boston, MA: Artech House.

8. List of Acronyms

AFRL	Air Force Research Laboratory
CF	Compress-and-Forward
CSI	Channel State Information
DBPSK	Differential Binary Phase Shift Keying
DF	Decode and Forward
DQPSK	Differential Quadrature Phase Shift Keying
DSA	Dynamic Spectrum Access
GMSK	Gaussian Minimum Shift Keying
MIMO	Multi-Input Multi-Output
MAC	Media Access Control
OTA	Over The Air
RF	Radio Frequency
RSSI	Received Signal Strength Indicators
SDR	Software Defined Radio
UAV	Unmanned Aerial Vehicle
USAFRL	United States Air Force Research Laboratory
USRP	Universal Software Radio Peripheral
VT	Virginia Tech